(2)

Technical Report 1408
April 1991

Iterated Transform Image Compression

Y. Fisher
E. W. Jacobs
R. D. Boss

NAVAL OCEAN SYSTEMS CENTER

San Diego, California 92152-5000

J. D. FONTANA, CAPT, USN
Commander

H. R. TALKINGTON, Acting
Technical Director

ADMINISTRATIVE INFORMATION

This work was conducted under the sponsorship of NOSC in-house Independent Exploratory Development Program (IED). The work was performed during fiscal year 1990 and funded under program element 0602936N. The work was performed by members of Code 633, Naval Ocean Systems Center, San Diego, CA 92152-5000.

Released by
J. C. Hicks, Head
Research Branch

Under authority of
R. H. Moore, Head
ASW Technology Division

MA

SUMMARY

OBJECTIVES

To present in a clear manner an algorithm based on iterated transforms that can be used to compress grayscale images. Demonstrate the algorithm and present results for various images.

RESULTS AND CONCLUSIONS

The theoretical framework for iterated transform image compression has been generalized to include noncontractive transforms. The method has been used to encode the 512 × 512 8-bpp image of "Lena" at a compression of 15.9:1 with the decode image having a root mean square error of 6.33 (32.1 dB). Other images have been encoded at various encoding conditions with the resulting compressions ranging from 10:1 to 63:1. It was shown that the relaxation of the contractivity constraint can lead to improvement in the fidelity of decoded images.

CONTENTS

FIGURES

CONTENTS (continued)

TABLES

1. INTRODUCTION

Lately, there has been much interest in the use of fractals to generate images. Barnsley and Sloan have popularized the converse notion of encoding images using fractals, see Barnsley (1988) and Barnsley and Sloan (1988). In this report, we present a scheme with a derivative dependence on fractals, which allows the encoding of monochrome images as a set of transformations. This leads to a resulting decrease in the memory required to store an image. The reconstructed image is an approximation of the original image, with a tradeoff between compression and fidelity. Although the current implementation of the encoding scheme is not superior to other lossy compression techniques, many possible improvements, which may lead to superior performance, have yet to be explored.

Hutchinson (1981) introduced the theory of iterated function systems (a term coined by Barnsley) to model self-similar sets (such as in figure 3). Demko, Hodges, and Naylor (1988) first suggested using iterated function systems to model complex objects in computer graphics. Barnsley, Demko, Elton, Sloan and others generalized the concepts and suggested the use of fractals to model "natural scenes." In his thesis, Jacquin (1989) described an image-encoding scheme based on iterated Markov operators on measure spaces and used it to encode 6-bit/pixel (bpp) monochrome images. This report presents a reformulation of this theory in a simplified and clarified manner. The theory has also been generalized to include noncontractive transforms. Eliminating the contractivity constraint for individual transforms has not previously been considered, and it is shown that it can result in a modest improvement in image quality.

2. A SIMPLE EXAMPLE

The example in this section serves as a simple illustration of the concepts involved in the image-encoding scheme presented later. This example is based on iterated function systems (IFS) which have been popularized by Barnsley (1988). Although the scheme presented later and iterated function systems have several features in common, the reader should be forewarned that they are not the same. The main concept is that the image of a set (a Sierpinski gasket, in this case) can be reconstructed

from a set of transformations which may take less memory to store than the original. As much mathematical formalism as possible has been excluded from this section, leaving this for section 3.

Consider the three transformations shown in figure 1. They are

$$w_1\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$w_2\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{2} \end{bmatrix}$$

$$w_3\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix} .$$

For any set S, let

$$W(S) = \bigcup_{i=1}^{3} w_i(S).$$

Denote the n-fold composition of W with itself as $W^{\circ n}$. Define $A_n = W(A_{n-1}) = W^{\circ n}(A_0)$ and arbitrarily choose A_0 as the unit square with lower left corner at the origin (i.e., $A_0 = \{(x,y)|0 \le x \le 1, 0 \le y \le 1\}$). Then as $n \to \infty$, the set A_n converges (in a sense not defined, but certainly visually) to a limit set A_∞. In fact, for any compact set $S \subset R^2$, $W^{\circ n}(S) \to A_\infty$ as $n \to \infty$. Figure 2 shows A_1, A_2, A_3 and A_4. Figure 3 shows the limit set A_∞.

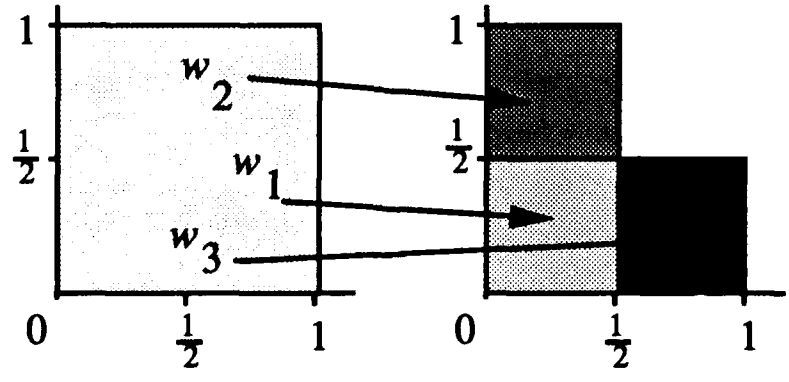


Figure 1. Three affine transformations in the plane.

That all compact initial sets converge under iteration to A_∞ is important—it means that the set A_∞ is defined by the w_i only. It is not difficult to see why this is

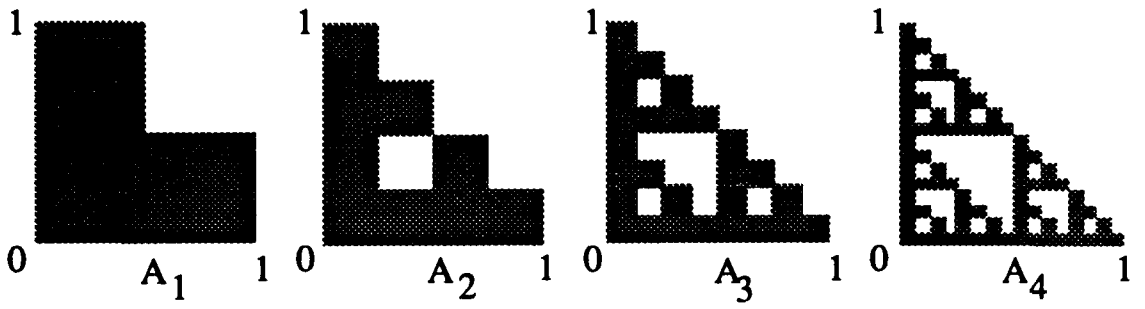


Figure 2. $A_1 = W(A_0)$ and its images A_2, A_3 and A_4.

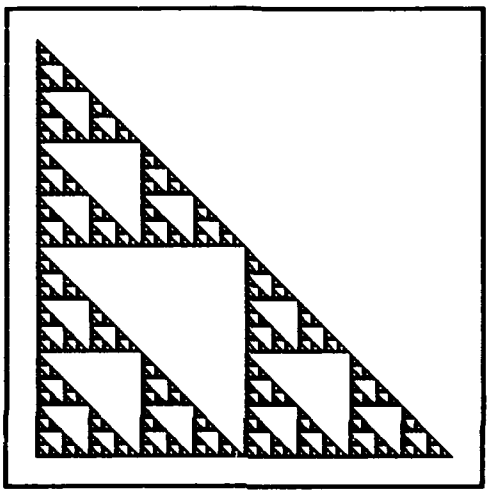


Figure 3. Limit set $A_\infty = \lim_{n\to\infty} W^{\circ n}(A_0)$.

true: The w_i are contractive (in this case they halve the diameter of any set to which they are applied). Thus, any initial A_0 will shrink to a point in the limit as the w_i are repeatedly applied.

Each w_i is determined by 6 real values, so that for this example 18 floating point numbers are required. In single precision, this requires 72 bytes. The memory required to store an image of the set depends on the resolution; figure 3 requires $256 \times 256 \times 1$ bit = 8192 bytes of memory. The resulting compression ratio in this example is 113.8.

In the example, the image of the Sierpinski gasket is described by a set of pixels, each being either black or white. It is inherently difficult to find an IFS which will encode an arbitrary set. The theory of IFS's has been extended by Barnsley and Jacquin (1988) to allow transforms to operate on only parts of the set rather than the entire set, in a method they call recurrent iterated function systems (RIFS). This extension can encode a larger set of images (Barnsley and Jacquin, 1988), and has

been used in a fully automated encoding system (Jacobs, Boss, and Fisher, 1990). The problem addressed in this report is the encoding of general monochrome images (i.e., an image in which each pixel has many possible gray levels, not just black or white). This type of image can be thought of as a three-dimensional object, each pixel having an x, y coordinate, and an intensity value z. To apply the basic concepts of RIFS to a three-dimensional image, one need only generalize the transformations to three dimensions.

3. THE MODEL

3.1 INTRODUCTION

This section contains a description of an iterative dynamical system (a space and a map from the space to itself) $W : F \to F$ used to model and encode images. The space F is a space of images, and the mapping W is a contraction. This ensures that the dynamical system has the most boring dynamics possible — rapid convergence to a fixed point. The goal is to construct the mapping W so that its fixed point is "close" to any given image that is to be encoded. Decoding then consists of iterating the mapping W from any initial image until the iterates converge to the fixed point.

Let $I = [0, 1]$ and I^n be the n-fold Cartesian product of I with itself. Let F be the space consisting of all graphs of real Lebesgue measurable functions $z = f(x, y)$ with $(x, y, f(x, y)) \in I^3$. Note that f is required to be bounded. A point in F can be thought of as an abstract image of infinite resolution, with $f(x, y)$ representing the gray level (with 0 being black and 1 being white) at the point (x, y) in the image. Images with finite resolutions can be modeled by partitioning I^2 with a rectilinear grid and either insisting that f be constant on the boxes of the grid, or by averaging f over each box. Color images can be encoded as graphs of functions $f : I^2 \to I^3$ with range points representing the color model of choice, for example RGB values.

Since the contractive mapping fixed point theorem requires a complete metric space, a carefully defined space and an associated metric is needed. The problem becomes one of finding a metric on F. Since the choice of metric in this paper serves mostly as motivation, the metric will be chosen to be as simple as possible:

$$\delta(f, g) = \sup_{x,y \in I} |f(x, y) - g(x, y)|. \tag{1}$$

The pair (F, δ) forms a complete metric space.

Other image models have been described, notably that of a positive measure over a Borel field (Jacquin, 1989). The argument for using this model is compelling: at any resolution viewing an image consists of sensing the flux of light through many small areas, for example the light that falls on each cone or rod in the retina. This flux can be naturally thought of as the measure of a small area in the image. We have opted not to adopt this model for the following reasons: First, the image model which is in fact manipulated, is not naturally a measure, but a function (which can be thought of as modeling a measure). Second, the space of measures is more difficult to metrize than F. In any case, this model has all the versatility of the measure model without the extraneous mathematical baggage.

The following sections describe two types of transformations $W : F \to F$ which will be used to encode and decode images. The first, called *z contractions* are simpler to define and generate than the second, called *eventual contractions.* Both types of transformations will define contractive maps, enabling the use of the contractive mapping fixed point theorem to find fixed points of W easily.

3.2 Z-CONTRACTIVE MAPPINGS

The precise statement of the mappings used requires several definitions.

Let $\pi_z : I^3 \to I^2$ be the projection operator defined by $\pi_z(x, y, z) = (x, y)$.

A map $w : \mathbf{R}^3 \to \mathbf{R}^3$ is said to be *z-contractive* with z-contractivity s if there exists a positive real number $s < 1$ such that for all $x, y, z_1, z_2 \in \mathbf{R}$,

$$|w(x, y, z_1) - w(x, y, z_2)| < s|z_1 - z_2|, \tag{2}$$

and $\pi_z \circ w(x, y, z)$ does not depend on z.

Let $D_1, \ldots, D_n$ be subsets of I^2 and $v_1, \ldots, v_n : I^3 \to I^3$ be some collection of maps. Define w_i as the restriction

$$w_i = v_i|_{D_i \times I}.$$

The maps $w_1, \ldots, w_n$ are said to *tile* I^2 if $\{\text{for all } f \in F, \bigcup_{i=1}^n w_i(f) \in F\}$. This means the following: for an image $f \in F$, each D_i defines a part of the image $f \cap (D_i \times I)$ to which w_i is restricted. When w_i is applied to this part, the result must be a graph

of a function over $R_i = \pi_z \circ w_i(f)$. It is also required that $I^2 = \cup_{i=1}^n R_i$, i.e., that the union $\cup_{i=1}^n w_i(f)$ yield a graph of a function over I^2. In particular, this means that the R_i are disjoint. See figure 4. The map W is defined as

$$W = \bigcup_{i=1}^{n} w_i. \tag{3}$$

Note that $W : F \to F$ is well defined only when the collection $w_1, \ldots, w_n$ tiles I^2.

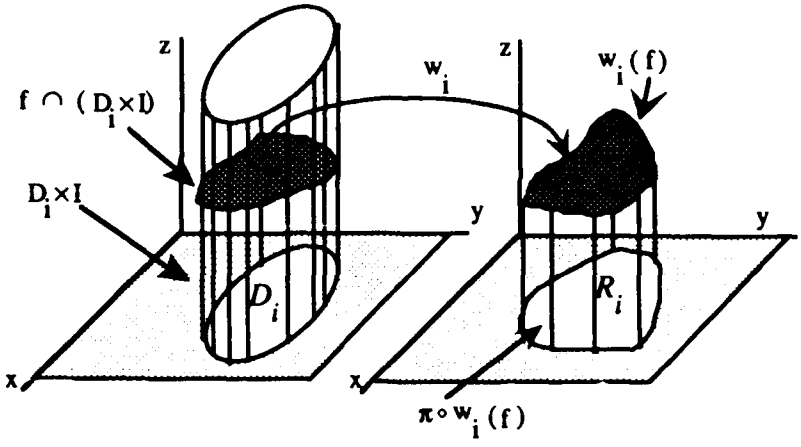


Figure 4. Parts of the tiling of an image.

During the rest of the discussion, it is assumed that $w_1, \ldots, w_n$ tile I^2 and that W is defined as in equation 3.

If there exists a positive $s < 1$ such that for any $f, g \in F$,

$$\delta(W(f), W(g)) < s\delta(f, g), \tag{4}$$

then W is called a *contraction* and s is called the *contractivity* of W. Note that W may be a contraction and still separate points in the x and y directions.

Claim 3.1 *If $w_1, \ldots, w_n$ are z contractions, then W is a contraction.*

Proof. Let $s_{1,\ldots,n}$ be the z contractivities for the transforms $w_{1,\ldots,n}$. For $f, g \in F$ and some s, $1 > s \geq \max_{i=1,\ldots,n}\{s_i\}$,

$$\delta(W(f), W(g)) \;=\; \sup\{|W(f(x,y)) - W(g(x,y))| \,\big|\, (x,y) \in I^2\}$$

$$
\begin{aligned}
\delta(W(f), W(g)) &= \sup\{\,|[w_i(x,y,f(x,y)) - w_i(x,y,g(x,y))] \cdot \hat{k}|\,\big| \\
&\qquad (x,y) \in D_i,\ i = 1,\ldots,n\} \\
&\quad \text{(where } \hat{k} \text{ is the unit vector in the } z \text{ direction)} \\
&\leq \sup\{s_i |f(x,y) - g(x,y)|\,\big|\, i = 1,\ldots,n\} \\
&\leq \sup\{s|f(x,y) - g(x,y)|\} \\
&\leq s \sup\{|f(x,y) - g(x,y)|\} \\
&\leq s\delta(f,g). \blacksquare
\end{aligned}
$$

For completeness, the following theorem is included, using the notation of this section.

Contractive Mapping Fixed Point Theorem. *Let F be a complete metric space with metric δ. If $W : F \to F$ is a contraction, then there exists a unique point $g \in F$ such that $g = W(g)$. Moreover, for any $f \in F$, the fixed point is the limit $g = \lim_{n\to\infty} W^{\circ n}(f)$.*

Following Hutchinson's notation (1981), the fixed point is denoted $|W| = g = \lim_{n\to\infty} W^{\circ n}(f)$. Then

$$
|W| = W(|W|) = \bigcup_{i=1}^{n} w_i(|W|). \tag{5}
$$

The transformation W is said to *encode an image* $f \in F$ if $f = |W|$. Given W, it is easy to find the image that it encodes—begin with any image f_0 and successively compute $W(f_0), W(W(f_0)), \ldots$ until the images converge to $|W|$. The converse is considerably more difficult: given an image f, how is a mapping W found such that $|W| = f$? There is no general, nontrivial solution to this problem. Instead an image $f' \in F$ can be found such that $\delta(f, f')$ is minimal with $f' = |W|$. Equation 5 suggests how this might be possible. Domains $D_1, \ldots, D_n$ are sought with corresponding transformations $w_1, \ldots, w_n$ such that

$$
W(f) = \bigcup_{i=1}^{n} w_i(f). \tag{6}
$$

This equation says: cover f with parts of itself; the parts are defined by the D_i and the way those parts cover f is determined by the w_i. Equality in equation 6 would imply that $f = |W|$. Since one cannot hope to exactly cover f with parts of itself, the optimal solution is sought, and then one hopes that $|W|$ and f will not look too different, i.e., that $\delta(|W|, f)$ is small. The following observation (Barnsley, 1988), known as the Collage Theorem, gives hope that this can be done. It is a corollary of the contractive mapping fixed point theorem.

Corollary 3.1 (Collage Theorem). *Let $W : F \to F$ be a contraction with contractivity s and let $f \in F$ be an image. Then*

$$\delta(|W|, f) \leq \frac{1}{1-s}\delta(W(f), f).$$

The problem is to find a W such that $\delta(W(f), f)$ is minimized and such that s is small. In that case, $|W|$ will be close (in δ) to f. However, as will be shown in section 5, the bound in the corollary is not very good; it provides motivation only and not a useful bound in practice. In fact, it is possible to generate examples in which the bound in the corollary is arbitrarily large while $\delta(|W|, f)$ is bounded. Empirical results (see table 2) show that restricting s to be small can result in a bound, but a less accurate reconstructed image.

It is always possible to approximate any given image of finite resolution f to within any $\epsilon > 0$. This can be done by simply mapping the whole image onto each pixel, for example. However, this is not a deep point, because compression is sacrificed in order to achieve accuracy; that is, a large number of maps $w_1, \ldots, w_n$ is required in order to have $\delta(f, W(f))$ small.

3.3 EVENTUALLY CONTRACTIVE MAPPINGS

This section contains a description of a more general class of transformations used to encode images. Those initiated to IFS theory (from which the example in section 2 is drawn) may find it surprising that when the transformations w_i are constructed, it is not necessary to impose any contractivity conditions on the individual transforms—not in the x, y, or z axis. In fact, for all sets of transformations given in this report (except for the two special cases given in table 2) the w_i's are not forced to be z contractive.

A map $W : F \to F$ is *eventually contractive* if there exists a positive integer m such that $W^{\circ m}$ is contractive. The exponent m is called the exponent of eventual contractivity. Note that for any set of transforms, for which an m exists, there is a minimum value of m. All contractive maps are eventually contractive, but not vice versa.

The following is a generalization of Corollary 3.1. As before, it is assumed that $w_1, \ldots, w_n$ tile I^2. The *z-contractivity* of w_i is the smallest number s satisfying equation 4, without the requirement that $s < 1$. Let

$$s_{max} = \max_{i=1,\ldots,n} \{s : s = \text{z-contractivity of } w_i\}.$$

Claim 3.2 *For $f \in F$ and $W : F \to F$ eventually contractive with minimum exponent of eventual-contractivity m and eventual-contractivity $\sigma < 1$,*

$$\delta(|W|, f) \leq \frac{1}{1-\sigma} \frac{1 - s_{max}^m}{1 - s_{max}} \delta(W(f), f).$$

Proof. The proof follows the same lines as the proof of corollary 3.1 (Barnsley, 1988). Since $W^{\circ m}$ has contractivity σ, Corollary 3.1 implies that

$$\delta(|W^{\circ m}|, f) \leq \frac{1}{1-\sigma} \delta(W^{\circ m}(f), f).$$

Also

$$\begin{aligned}
\delta(W^{\circ m}(f), f) &\leq \delta(W^{\circ m}(f), W^{\circ m-1}(f)) + \delta(W^{\circ m-1}(f), f) \\
&\leq \sum_{i=1}^{m} \delta(W^{\circ i}(f), W^{\circ i-1}(f)) \\
&\leq \left(\sum_{i=1}^{m} s_{max}^{i-1} \right) \delta(W(f), f) \\
&\leq \frac{1 - s_{max}^m}{1 - s_{max}} \delta(W(f), f).
\end{aligned}$$

It remains to show that $|W| = |W^{\circ m}|$. This follows since $|W^{\circ m}| = \lim_{n \to \infty} W^{\circ n}(g)$ is independent of g. ∎

With some extra notation, it is possible to improve this estimate, but since this claim and Corollary 3.1 provide motivation, not serious bounds, it serves little purpose to do so.

A brief explanation of how a transformation $W : F \to F$ can be eventually contractive but not z contractive is in order. The map W is composed of a union of maps w_i operating on disjoint parts of an image. If any of the w_i are not z contractive, then W will also not be contractive. The iterated transform $W^{\circ m}$ is composed of a union of compositions of the form

$$w_{i_1} \circ w_{i_2} \circ \cdots w_{i_m}.$$

Since the contractivities multiply to yield the contractivity of the composition, the compositions may be contractive if each contains sufficiently contractive w_{i_j}. Thus W will be eventually contractive if it contains sufficient "mixing" so that the contractive w_i eventually dominate the expansive ones. In practice, this condition is relatively simple to check.

4. THE IMPLEMENTATION

This section describes an implementation of the image compression algorithm described in the previous sections. Many features of this implementation are similar to that of Jacquin (1989). The differences will be examined in the discussion section.

In brief, the encoding of an image can be described as follows. Recall that the w_i are maps into I^3 and that $R_i, D_i \subset I^2$. The R_i will be called ranges and the D_i domains, even though they are not the domains and ranges of the w_i. Nevertheless, the terminology is commonly used.

To encode an image f, w_i and D_i which tile I^2 must be found such that W (defined by equation 3) is contractive or eventually contractive. Since the goal is to limit the memory required to specify W, I^2 is partitioned by geometrically simple sets R_i with $\cup_{i=1}^{n} R_i = I^2$. For each R_i, a $D_i \subset I^2$ and $w_i : D_i \times I \rightarrow I^3$ is sought such that $w_i(f)$ is as δ close to $f \cap (R_i \times I)$ as possible; that is,

$$\delta(f \cap (R_i \times I), w_i(f)) \tag{7}$$

is minimized. To limit the memory required to specify w_i, only maps of the form

$$w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix} \tag{8}$$

are considered, where w_i is restricted to $D_i \times I$. This constrains the sets D_i as well, since they must (after projection by π_z) map onto the R_i. In fact, as will be described in more detail in the following paragraphs, further restrictions are placed upon the possible values for the coefficients in equation 8, thus resulting in a compact specification for the w_i.

There are two fairly independent considerations for implementing the encoding algorithm. First, the set **D** of all possible domains from which the D_i's are chosen to encode the image must be defined. Similarly, the set **R** of all possible ranges from which the R_i's are chosen to encode the image must be defined. Although, in general, **R** and **D** can be chosen to be any collection of subsets of I^2, in this implementation they are taken to be collections of squares only. The choice of **R** and **D** limits the amount of information required to specify the geometry of the sets thereby increasing the resulting compression. This choice also simplifies many of the computations. On the other hand, it severely restricts the encoding process, in the sense that it becomes more difficult to collage an image by transformations of parts of itself.

For 256×256 pixel images the model of section 3 is scaled to $[0,255] \times [0,255] \times [0,255]$. (For other image sizes, appropriate scaling was employed.) The set $\mathbf{R}$ is chosen to consist of 4×4, 8×8, 16×16, and 32×32 nonoverlapping subsquares of $[0,255] \times [0,255]$. The collection $\mathbf{D}$ consists of 8×8, 16×16, 24×24, 32×32, 48×48, and 64×64 subsquares with sides which are parallel to or slanted at 45 degree angles from the natural edges of the image. To reduce the amount of information required to specify a particular domain square, domain squares of size $s \times s$ are restricted to be centered on a lattice with vertical and horizontal spacing of $s/2$.

Given a range square R_i and a domain square D_i, there are eight possible orientations for D_i to have when mapped onto R_i by a map of the form of equation 8. These

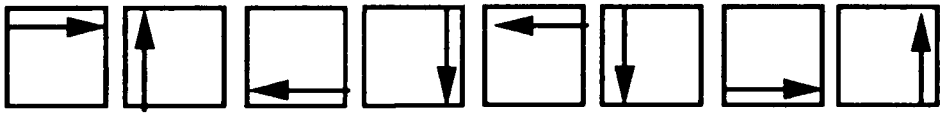


Figure 5. Eight symmetries of the square.

orientations are shown in figure 5. The size and position of R_i and D_i, and a given orientation define the coefficients a_i, b_i, c_i, d_i, e_i, and f_i in equation 8. Insisting that w_i map (the graph above) D_i to (a graph above) R_i while minimizing equation 7, determines s_i and o_i. In this way w_i is determined uniquely for a chosen metric. In the examples in section 5, the root mean square (rms) error (δ_{rms}) was chosen as the metric. This choice was made for ease of implementation and the fact that δ_{rms} is more reflective of visual accuracy than δ. However, δ_{rms} is not an ideal measure of image fidelity, and it may be that among the various L_p norms, there is a better choice. The proofs in section 3 do not hold for the root mean square metric. This does not raise a serious objection to the use of δ_{rms}, since when the iterates of a transformation converge in δ, they converge in δ_{rms}.

Once the choice of $\mathbf{R}$ and $\mathbf{D}$ is made, the encoding problem is reduced to choosing a good set $\{R_i\} \subset \mathbf{R}$, and the corresponding set $\{D_i\} \subset \mathbf{D}$, such that good compression and an accurate encoding of the image results. The number of transformations is exactly the number of R_i's; therefore, the compression is inversely proportional to the number of R_i's used to tile the image. To take advantage of local "flatness" in the image and to reduce the error in regions of high variability, a recursive quadtree partitioning method is used to allow the range squares to vary in size depending on the local conditions in the image.

The choice of D_i's affects the accuracy of the image, and the method used to

find the D_i's determines how much computation time the encoding takes. A search through all of **D** would clearly result in the choice that would best minimize equation 7, but for applications where encoding time is a consideration, such a search may require too much computation time. To overcome this problem, a classification scheme is used to classify all possible domains in advance. The current range square is classified using the same scheme, and a search for the optimal domain square in the same class (or similar classes) is performed. If the best domain square and its corresponding w result in an error less than a predetermined tolerance, they are stored and the process is repeated for the next range square. If the predetermined tolerance is not satisfied, the range square is subdivided into four equal squares, and the process is repeated until the tolerance condition is satisfied, or a range square of the minimum size is reached.

Initially, the range squares R_i were chosen to be the 64 32×32 subsquares in an image. The minimum size allowed for the range squares was usually 4×4 (although, as mentioned below, sometimes this was increased to 8×8). For each range square tested, a domain square with side lengths greater than the side lengths of the range square was sought, such that the condition of equation 7 was minimized.

The domain and range squares were classified in the following way. Each square was divided into quadrants which were ordered from the brightest average intensity to the darkest average intensity. A symmetry operation consisting of rotations and flips was applied to bring the brightest, second brightest, and third brightest quadrants into one of the three canonical positions shown in figure 6. Once divided into these

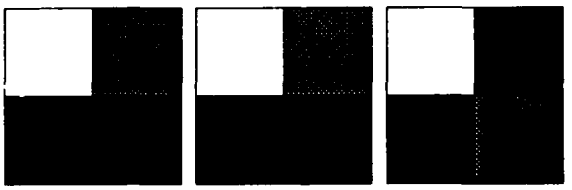


Figure 6. Three canonical orientations for square partitions.

three major classes, the quadrants of each square were ordered from most edge-like to least edge-like. This ordering results in $4! = 24$ possible symmetries, for a total of 72 classes.

The motivation for the classification scheme is simple. By orienting a square into one of the canonical positions, a symmetry operation is determined that is likely to be optimal (in the sense of minimizing equation 7). When searching for domains during

the encoding process, the symmetry operation determined by the classification scheme is used (the other seven possible orientations are not tested). The further classification by edge-like character is an attempt to maintain edge fidelity, since visual perception of image quality is sensitive to edge integrity.

Before discussing the actual results, it is necessary to describe in more detail the information that needs to be stored to determine the transformations:

- Size of the range square R_i (the position of R_i can be implicitly determined by the order of storage),
- Size, position, and orientation (i.e., 0 or 45 degrees) of the domain square D_i,
- Symmetry operation,
- Scale factor (s), and
- Offset (o).

The restriction on W to be *eventually contractive*, places no *a priori* limitation on the z-scaling s and the z-offset o. However, s and o must be stored using some fixed number of bits. Therefore, the restriction $0.05 < |s| < 2$ or $s = 0$ was used (this set can be stored to within 6 percent accuracy by 7 bits). The z-offset o was stored using 8 bits and could take the value of every fourth integer from -512 and 508. When the remaining data were stored efficiently, the average storage requirement per transformation was approximately 31 bits.

To decode an encoded image, any initial $f_0 \in F$ is chosen and the map $W : F \rightarrow F$ is iterated until the iterates converge to a fixed point $|W|$. The fixed point is the decoded image.

5. RESULTS

This section presents some results from this particular implementation. Table 1 summarizes the results for the figures shown. In the table, "$\delta(|W|, f)$" is the rms error per pixel, "N_D" is the average number of domain squares searched for each range square, "Time" is the cpu seconds required for the encoding on a Convex C210, "n" is the number of transformation in the encoding, "Size" is the size of the original figure

(and the decoded result), and "Comp" is the resulting compression for the figures. The values of the time are shown for comparison between the various encodings. The times have since been reduced dramatically by optimizing the encoding program.

Table 1. Results for the figures 8, 10, and 11d.

Figure	$\delta(\|W\|, f)$	N_D	Time	n	Size	Comp
8a	14.05	3035	1103	277	256	63.0
8b	8.73	4270	2797	601	256	28.2
8c	7.68	11340	5453	1003	256	16.6
8d	8.12	1472	738	1057	256	15.8
10	6.33	55184	96485	3949	512	15.9
11d	8.59	14155	9591	1654	256	10.0

Figure 7 is the original 256×256 8-bpp image of a dog. Figure 8 shows four images reconstructed from encodings of figure 7. In figures 8a and 8b, the lowest level of the quadtree partitioning was 8×8 squares (rather than 4×4 squares). Figures 8c and 8d have comparable compression and demonstrate the effect of the classification scheme on computation time. The average number of domain squares searched per range is determined by the number of classes searched when encoding. Searching through more classes does yield a better encoding but, as can be seen from table 1, results in longer computation time.

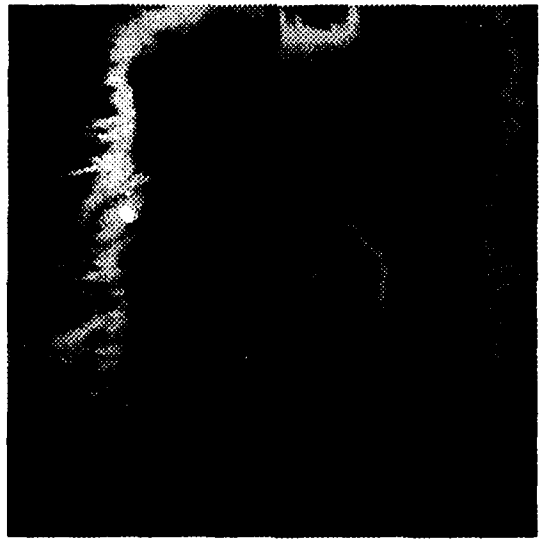


Figure 7. Original figure of the dog.

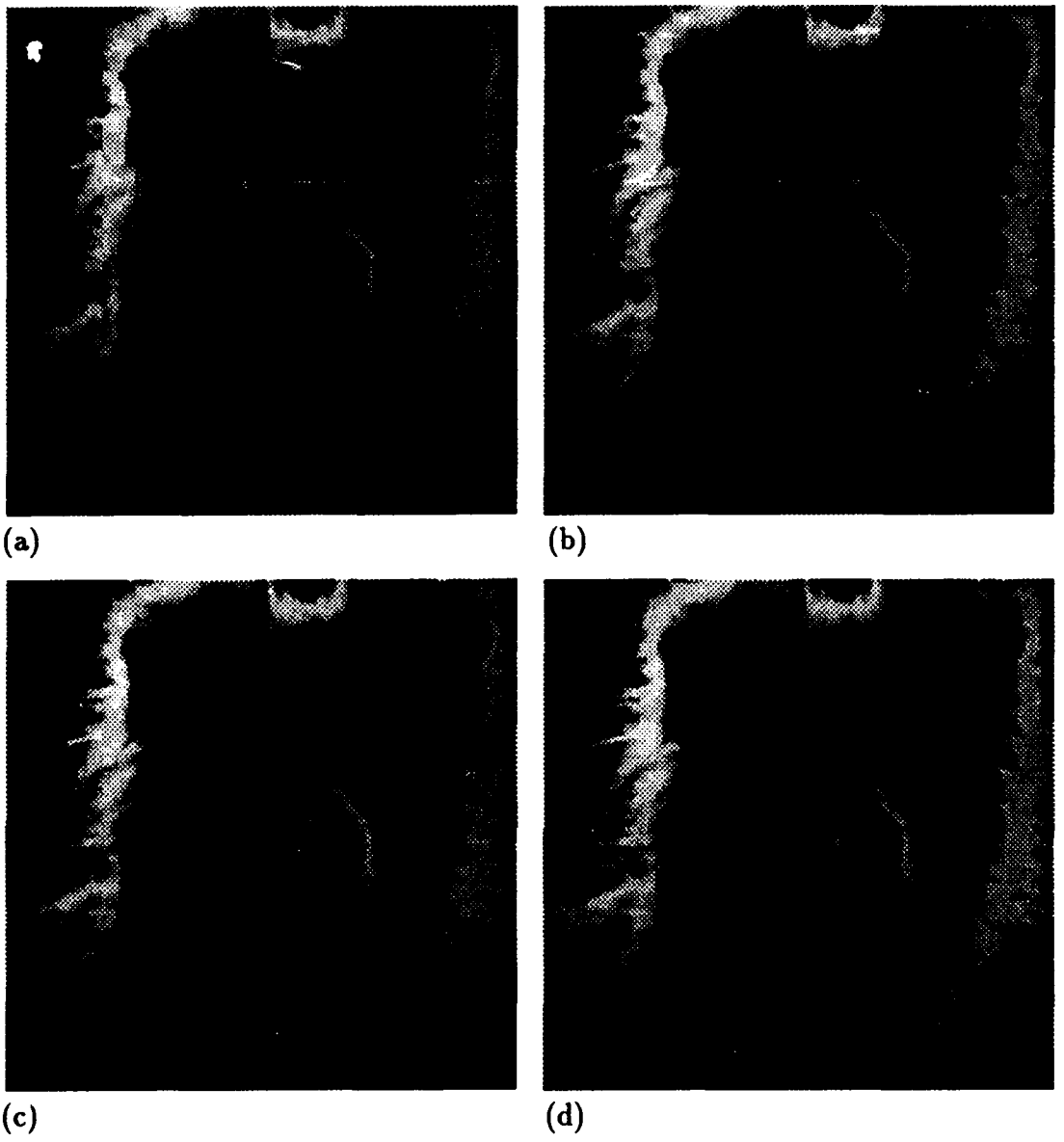


Figure 8. Four reconstructed images of the image of the dog. The compressions are (a) 63.0:1, (b) 28.2:1, (c) 16.6:1, and (d) 15.8:1.

Figure 9 shows the commonly used 512×512 image of Lena. Figure 10 shows a reconstructed Lena with compression 15.9:1 and an rms error per pixel of 6.33 (32.1 dB signal-to-noise ratio). This image has also been compressed at 35.9:1 with an rms error per pixel of 9.22 (28.8 dB).

Figure 9. Original 512×512 image of Lena.

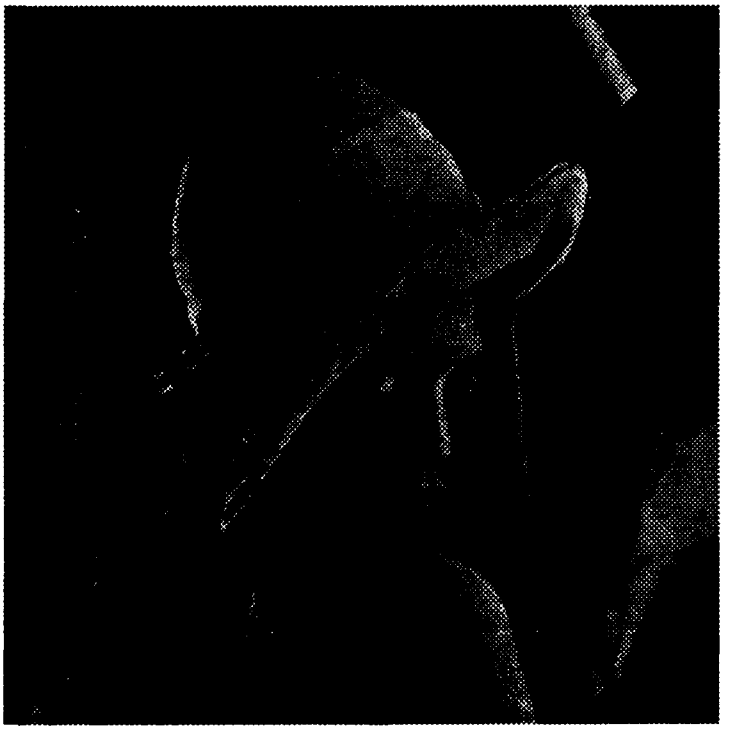


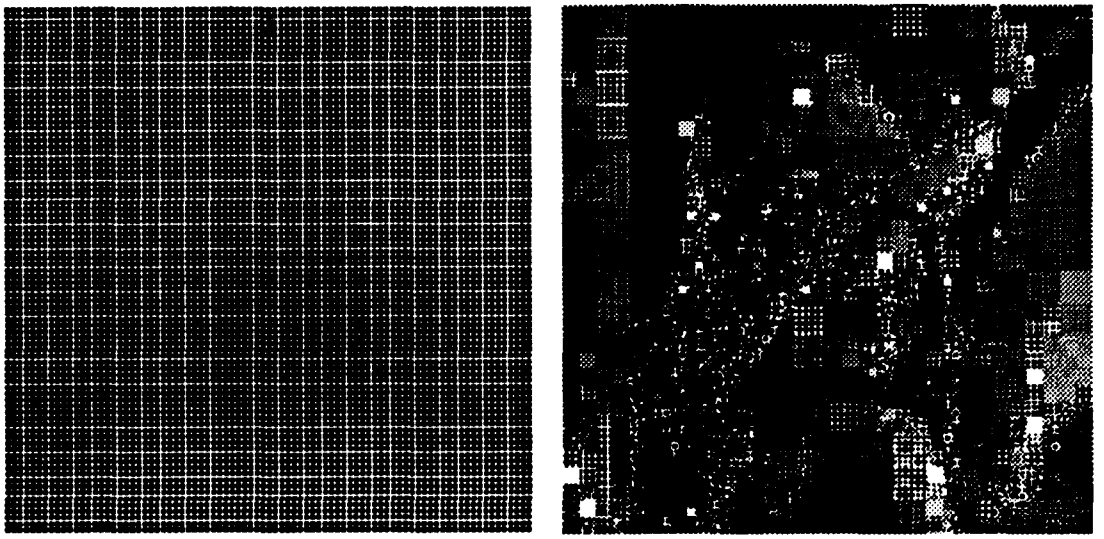
Figure 10. Decoded Lena at 15.9:1 compression.

6. DISCUSSION

For illustrative purposes, the decoding process for the encoded image Lena is shown in figure 11. This encoding was performed on a 256×256 original version of the image. Figure 11a is the initial image f. The grid pattern was chosen to illustrate the nature of the mappings in the subsequent figures. This choice is, of course, arbitrary. Figure 11b is the first iterate $W(f)$; figure 11c is the second iterate $W^{\circ 2}(f)$; and figure 11d is the tenth iterate $W^{\circ 10}(f) \approx |W|$ which approximates the fixed point of the system to within the resolution of the image. For all of the test images shown, decoding requires a comparable number of iterations. This is of particular interest because, as discussed in section 3, the w_i's are not restricted to be z contractive. The distribution of the scale factors used to encode the image of Lena (figure 11d) is shown in figure 12. It illustrates that there are a significant number of transformations that are not z contractive. Note that to maintain a constant accuracy, the possible values of the scale factor that are larger in magnitude are sparser.

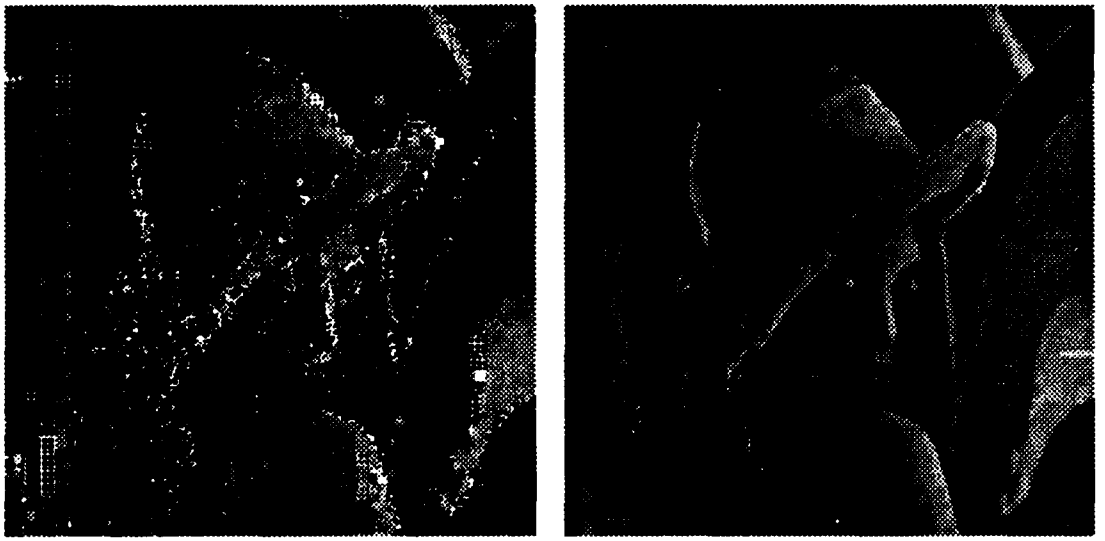
Since the relaxation of z contractivity has not been previously used, it is worthy of further examination. Let f be the 256×256 8-bpp pixel image image shown in figure 13a. Let $R_1, \ldots, R_{1024}$ be the 1024 nonintersecting 8×8 subsquares of $[0, 255] \times [0, 255]$. Let $\mathbf{D}$ be the collection of all 16×16 subsquares of $[0, 255] \times [0, 255]$ whose edges coincide with the edges of the R_i. This choice of $\mathbf{R}$ and $\mathbf{D}$ is simpler than that used for other results in this report. For each R_i, find the best $D \in \mathbf{D}$, denoted D_i, and a w_i of the form of equation 8 such that $\pi_z(w_i(f)) = R_i$ and such that equation 7 is minimized. In this example, D_i and w_i were chosen by searching through all possible choices for the pair that minimizes equation 8. To ensure that $\lim_{m \to \infty} W^{\circ m}$ exists, the w_i must be z contractive or W must be eventually contractive. In this example, the scale factor was not restricted to be between ± 2 as in other results in this report. Rather, s was restricted to be between ± 0.7 or ± 0.9 (two z-contractive mappings), or s had no restriction (this resulted in an eventually contractive mapping with maximum "z contractivity" $= 4.05$). Figure 13b shows the reconstructed image from the eventually contractive mapping.

Table 2 summarizes some results for this example. In the table, σ is the contractivity of $W^{\circ m}$, m is an exponent of eventual contractivity, n is the number of maps w_i, and δ-bound is the bound on $\delta(|W|, f)$ given by claim 3.2 or corollary 3.1. Because δ_{rms} is a better measure of the visual accuracy of an encoding than δ; it is used for $\delta_{rms}(W(f), f)$ and $\delta_{rms}(|W|, f)$. The bound in table 2 is given as δ because that is the metric for which corollary 3.1 and claim 3.2 are valid.



(a) An initial image f.

(b) The first iterate $W(f)$.



(c) The second iterate $W^{\circ 2}(f)$.

(d) The 10th iterate $W^{\circ 10}(f)$.

Figure 11. Decoding process for Lena.

Table 2. Results for encodings of figure 13a using different constraints on the allowable scale factors.

$\delta_{rms}(W(f), f)$	$\delta_{rms}(\|W\|, f)$	s_{max}	σ	m	n	δ-bound
21.96	23.49	0.7	0.7	1	1024	630
20.31	20.98	0.9	0.9	1	1024	1,740
18.94	19.62	4.05	0.85	5	1024	428,310

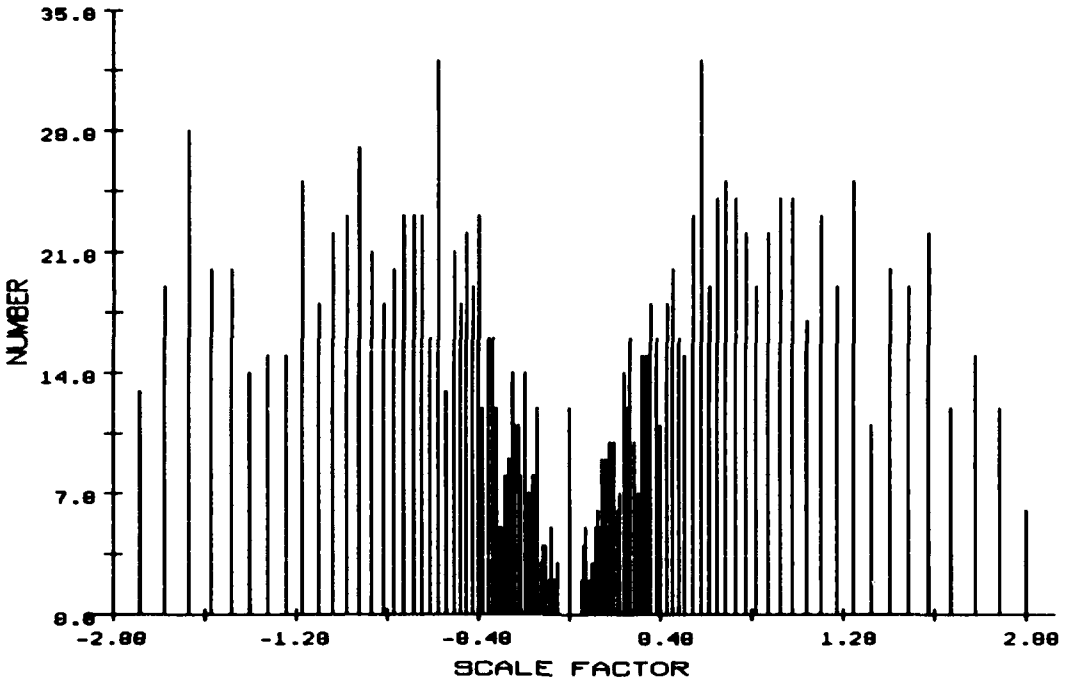


Figure 12. Distribution of z-scale factors in the encoding of Lena.

(a) (b)

Figure 13. (a) A sample image of Mara and (b) the fixed point for the encoded Mara.

The first two entries in table 2 are data for encodings resulting from z-contractive maps. The third entry is an encoding with an eventually contractive map. The table demonstrates several points. The most relevant point is that the eventually contractive map results in a moderately more accurate encoding. It is interesting to note that the difference between $\delta(W(f), f)$ and $\delta(|W|, f)$ for the eventually contractive mapping is not greater than this distance for the z-contractive mappings. The table also demonstrates that, even though restricting the z-contractivity of the w_i's (from 0.9 to 0.7 in the z-contractive mappings) improves the bound from corollary 3.1, it worsens the fidelity of the reconstructed image. It is evident that the bound given by claim 3.2 for the eventually contractive map is also poor. This demonstrates that the bounds in corollary 3.1 and claim 3.2 should be viewed as motivation rather than actual bounds.

While the use of noncontractive transforms does improve image fidelity, it is also true that to store the transforms requires more bits for those encodings in which the transforms are not limited to be contractive. In the quadtree implementation it is not necessarily true that the addition of bits to each transform must result in a loss of compression. This is because the additional scale factors may allow for a larger number of bigger ranges to be covered successfully. This issue of fidelity *vs.* compression is still under investigation.

Another point of interest is the distribution of distances between R_i and D_i. If this distribution indicates that a disproportional number of domain squares are relatively near to their range squares, then it might be advantageous to limit the set **D** to a localized area around each R_i. This probability distribution measures the degree of local self-similarity within an image. Figure 14 shows this distribution for the encoded image of Lena (figure 11d). The solid curve is the theoretical distribution of the distance between two randomly chosen points in I^2. The figure shows that there is no significant local self similarity; the slight shift between the theoretical and experimental distribution is due in part to the distance being measured between small but not infinitesimal squares. The distribution is typical for the test images studied.

It has already been pointed out that the relaxation of z contractivity distinguishes the data given here from other implementations of iterated transform image encoding. The data in figures 12 and 14 point out other major differences between this implementation and that of Jacquin (1989). Both the number of possible domains in **D** and the allowed values of the scale factor s are far more numerous in the implementation described here. As a result, this implementation will allow (roughly) only 2/3 the number of transformations allowed in Jacquin's scheme at the same

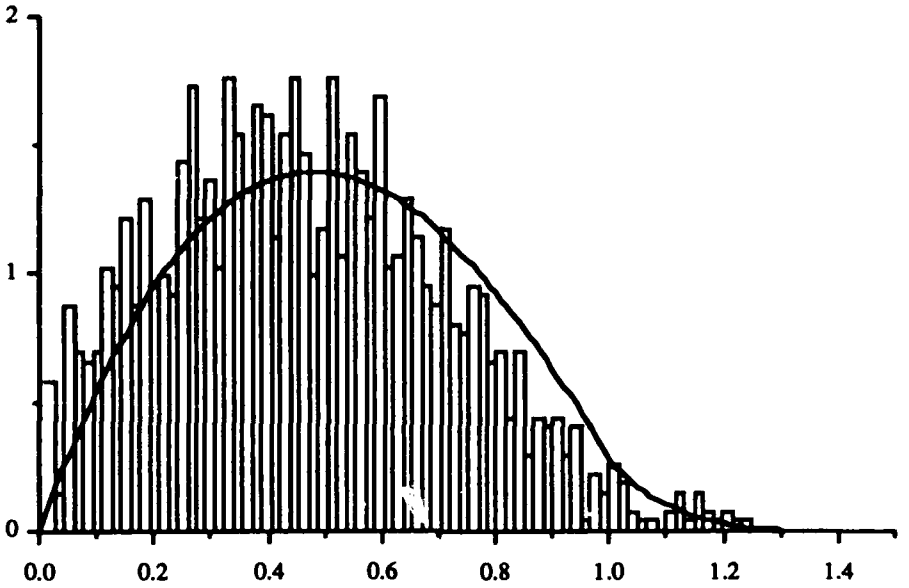


Figure 14. Theoretical distribution for random domain-range distance and the actual distribution for a typical encoding.

compression. Generalizing the transformations increases the storage requirement per transformation but also increases the possibility of covering larger range squares, thereby reducing the total number of transformations needed to encode an image. Clearly, one could imagine other schemes that would further increase (or decrease) the storage requirement per transformation. A systematic investigation of the dependence of system performance on this requirement has yet to be done. It should be noted that the performance of the implementation presented here, and that of Jacquin are quite similar (as measured by the rms error at the same compression). This is of interest because the approach of the two implementations are quite different, Jacquin using a large number of transformations from a small pool, and the implementation here using relatively few transformations from a large pool.

The size of the domain pool also effects the time necessary to search through the pool. As described in section 4, the current implementation makes use of a classification scheme to reduce the number of domains that need to be tested. Although the classification scheme presented results in classifications that are sufficient to obtain good encodings (and is the best of several tested), the data in table 1 for figures 8c and d show that a search through more classes results in some minor improvement. Clearly, more work could be done to find better classification schemes.

It should also be noted that with some postprocessing, such as smoothing the image at the boundary of the range squares, it is possible to greatly reduce the "box"

artifacts arising from the encoding process. Such postprocessing can also decrease the total rms error of the encoding. No postprocessing on any of the images presented in this paper was done.

Finally, an encoded image can be decoded at a larger size. Because the transformations can naturally create detail at all scales, such an enlarged decoding will not appear pixelized. In a certain sense, the resulting compression is increased, since now a larger image is decoded from the same information. The automatic creation of detail may be useful, but all of the images in this paper were decoded at the same size as the original.

7. REFERENCES

Barnsley, M.F. 1988. *Fractals Everywhere*, Academic Press, Inc., San Diego, CA.

Barnsley, M.F. and A.E. Jacquin. 1988. "Application of Recurrent Iterated Function Systems to Images," *SPIE vol. 1001, Visual Comm. and Image Processing*, p. 122.

Barnsley, M. F. and A.D. Sloan. 1988. *Byte*, vol. 13, p. 215.

Demko, S., L. Hodges, and B. Naylor. 1988. *ACM*, vol. 19, p. 3.

Hutchinson, J. E. 1981. "Fractals and Self-Similarity," *Indiana University Mathematics Journal*, vol. 35, p. 5.

Jacobs, E.W., R.D. Boss, and Y. Fisher. 1990. "Fractal-Based Image Compression, II," NOSC TR-1362, Naval Ocean Systems Center, San Diego, CA.

Jacquin, A. 1989. "*A Fractal Theory of Iterated Markov Operators, with Applications to Digitial Image Coding*," Ph.D. Thesis, Department of Mathematics, Georgia Institute of Technology.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	April 1991	Final: Sep 89 – Oct 90

4. TITLE AND SUBTITLE

ITERATED TRANSFORM IMAGE COMPRESSION

5. FUNDING NUMBERS

PE: 0602936N
PROJ: RV36121
SUBPROJ: 63-ZE88-01
WU: IC000037

6. AUTHOR(S)

Y. Fisher, E. W. Jacobs, R. D. Boss

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Ocean Systems Center
San Diego, CA 95152–5000

8. PERFORMING ORGANIZATION REPORT NUMBER

NOSC TR 1408

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Ocean Systems Center
Independent Exploratory Development Program (IED)
San Diego, CA 95152–5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

In-house

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Preliminary results obtained from a lossy automatic image compression scheme based on iterated transforms are presented. A theoretical image model based on iterated mapping of functions is presented. The model does not require strict contractivity for the individual transformations. Relaxation of the contractivity requirement empirically has resulted in better encodings. The 512 × 512 8-bpp image of "Lena" was compressed at 15.9:1 with a resulting root mean square error per pixel of 6.33 (32.1 dB). Other images have been encoded using various encoding conditions with the resulting compressions ranging from 10:1 to 63:1.

14. SUBJECT TERMS

fractals, iterated function systems (IFS), image compression, eventually contractive mappings, bits per pixel (bpp)

15. NUMBER OF PAGES

31

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

INITIAL DISTRIBUTION

Code 0012	Patent Counsel	(1)
Code 014	W. T. Rasmussen	(1)
Code 0141	A. Gordon	(1)
Code 0142	K. J. Campbell	(1)
Code 0144	R. November	(1)
Code 60	F. E. Gordon	(1)
Code 63	R. H. Moore	(1)
Code 633	J. C. Hicks	(1)
Code 633	E. W. Jacobs	(50)
Code 952B	J. Puleo	(1)
Code 961	Archive/Stock	(6)
Code 964B	Library	(3)

Defense Technical Information Center
Alexandria, VA 22304-6145 (4)

NOSC Liaison Office
Washington, DC 20363-5100 (1)

Center for Naval Analyses
Alexandria, VA 22302-0268 (1)